AMENDMENTS TO THE CLAIMS

Please amend the claims as indicated in the following listing of all claims:

1. - 20. Canceled

21. (Previously Presented) A method comprising:

speculatively locking a resource to be accessed by execution of a first instruction, wherein the locking is performed prior to determining whether a hazard exists between the access and execution of a second instruction.

22. (Previously Presented) The method of claim 21 wherein the locking is performed prior to the first instruction entering a trap stage of an instruction pipeline.

23. (Previously Presented) The method of claim 21 wherein the first instruction is an atomic instruction including a portion to lock the resource and a portion to unlock the resource.

24. (Previously Presented) The method of claim 21 wherein the hazard includes a read-after-write hazard.

25. (Previously Presented) The method of claim 21 wherein the locking includes:

locking the resource during an effective address calculation stage of an instruction pipeline.

26. (Previously Presented) The method of claim 21 wherein the locking includes locking at least a portion of a cache.

27. (Previously Presented) The method of claim 21 wherein the locking includes locking at least one memory address.

28. (Previously Presented) The method of claim 21 further comprising unlocking the resource no later than a time at which the first instruction exits an instruction pipeline, regardless of whether the first instruction is cancelled.

29. (Previously Presented) The method of claim 28 wherein unlocking the resource includes:

unlocking the resource in the normal course of executing the computer instruction.

30. (Previously Presented) The method of claim 28 wherein unlocking the resource includes:

preventing a write portion of the first instruction from altering information held in at least a portion of the resource.

31. (Previously Presented) The method of claim 30 wherein preventing a write portion from altering information includes suppressing writing a value to an architectural storage location.

32. (Previously Presented) A processor comprising:

at least one processing core to speculatively lock a resource in response to an access executed by a first instruction prior to determining whether a hazard exists between the access and execution of a second instruction.

33. (Previously Presented) The processor of claim 32 further comprising a plurality of processing cores, wherein respective processing cores are adapted to lock the resource in response to respective accesses by respective first instructions prior to determining whether a hazard exists between the respective accesses and the second instruction.

34. (Previously Presented) The processor of claim 32 wherein said at least one processing core is adapted to lock the resource prior to the first instruction entering a trap stage of a pipeline.

35. (Previously Presented) The processor of claim 32 wherein said at least one processing core is adapted to implement an atomic instruction, wherein implementing the atomic instruction includes locking the resource and unlocking the resource.

36. (Previously Presented) The processor of claim 32 wherein said processing core locks the resource before it is determined if a read-after-write hazard exists.

37. (Previously Presented) The processor of claim 32 wherein said processing core locks the resource during an effective address calculation stage of a pipeline.

38. (Previously Presented) The processor of claim 32 further including a cache, and wherein locking a resource includes locking at least a portion of the cache.

39. (Previously Presented) The processor of claim 32 wherein said processing core further includes an output coupled to a memory, and wherein locking a resource includes locking at least one memory address.

40. (Previously Presented) The processor of claim 32 further comprising logic to unlock the resource no later than a time at which the first instruction exits an instruction pipeline, regardless of whether the first instruction is cancelled.

41. (Previously Presented) The processor of claim 40 wherein the processor includes logic to prevent a write portion of the first instruction from altering information held in at least a portion of the resource if the first instruction is cancelled.

42. (Previously Presented) A processor adapted to speculatively dispatch a load operation to a cache unit prior to determining whether read-after-write hazards associated with the load operation are present.

43. (Previously Presented) The processor of claim 42 wherein the processor is adapted to lock a resource associated with the load operation concurrently with dispatching the load operation.

44. (Previously Presented) The processor of claim 43 wherein the processor is further adapted to unlock the resource associated with the load operation no later than a time at which an

instruction implementing the load operation exits an instruction pipeline, regardless of whether the instruction is cancelled before exiting the instruction pipeline.

45. (Previously Presented) A processor comprising:

means for determining whether a hazard exists between an access to a resource to be performed by a first instruction and execution of a second instruction; and

means for locking the resource prior to determining whether the hazard exists.

46. (Previously Presented) The processor of claim 45 wherein the locking means includes means for locking the resource prior to the first instruction entering a trap stage of an instruction pipeline.

47. (Previously Presented) The processor of claim 45 wherein the first instruction is an atomic instruction including a portion to lock the resource and a portion to unlock the resource.

48. (Previously Presented) The processor of claim 45 wherein the determining means includes means for determining whether a read-after-write hazard exists.

49. (Previously Presented) The processor of claim 45 wherein the locking means includes:

means for locking the resource during an effective address calculation stage of an instruction pipeline.

50. (Previously Presented) The processor of claim 45 wherein the locking means includes means for locking at least a portion of a cache.

51. (Previously Presented) The processor of claim 45 wherein the locking means includes means for locking at least one memory address.

52. (Previously Presented) The processor of claim 45 further comprising means for unlocking the resource no later than a time at which the first instruction exits an instruction pipeline, regardless of whether the first instruction is cancelled.

53. (Previously Presented) The processor of claim 52 wherein the unlocking means includes:

means for unlocking the resource in the normal course of executing the first instruction.

54. (Previously Presented) The processor of claim 52 wherein the unlocking means includes:

means for preventing a write portion of the first instruction from altering information held in at least a portion of the resource.

55. (Previously Presented) The processor of claim 54 wherein the preventing means includes means for suppressing writing of a value to an architectural storage location.